从基础到实践 基础知识全面覆盖 实践操作循序渐进

从必需到拓展 理论讲解详尽具体 动手应用实操实练

从入门到进阶 内容编排由浅入深 进阶案例综合拓展

重点推荐

JSP设计与开发（第3版）

陈磊 徐受蓉◎主编

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

JSP 设计与开发

(第 3 版)

主　编　陈　磊　徐受蓉

副主编　尚　晋　董　明

赵叶青　刁　绫

参　编　冯　林　蒋文豪

舒　蕾　罗少甫

主　审　桑　军

北京理工大学出版社

BEIJING INSTITUTE OF TECHNOLOGY PRESS

内 容 简 介

本教材详细介绍了基于 Java 的 Web 开发所需的基础知识和技术,主要内容包括 JSP 概述、Web 开发基础、JSP 语法基础、JSP 内置对象、JDBC 技术、JavaBean 技术、Servlet 技术、标准标签库 JSTL、Struts 应用、Spring 框架应用、Ajax 技术应用、学生课绩管理系统等。

教材根据 Java Web 程序员的岗位能力要求和学生的认知规律组织内容。通过 56 个完整的案例(案例视频可通过访问课程网站或扫描书中二维码直接观看),系统地介绍了 JSP 设计与开发所涵盖的技术,将理论知识介绍和实践技能训练有机结合,“教、学、做”一体,适合理实一体化的教学模式。同时,在该课程的精品课程网站上提供了完备的教学资源。

本书可作为计算机类专业的教材,也可以作为计算机培训班的教材,以及 Web 程序员的参考书。

图书在版编目(CIP)数据

JSP 设计与开发/陈磊, 徐受蓉主编 . —3 版 . —北京:北京理工大学出版社,2019. 11(2020. 6 重印)

ISBN 978 – 7 – 5682 – 7849 – 2

Ⅰ. ①J…　Ⅱ. ①陈…　②徐…　Ⅲ. ①JAVA 语言 – 网页制作工具 – 教材　Ⅳ. ①TP312. 8　②TP393. 092. 2

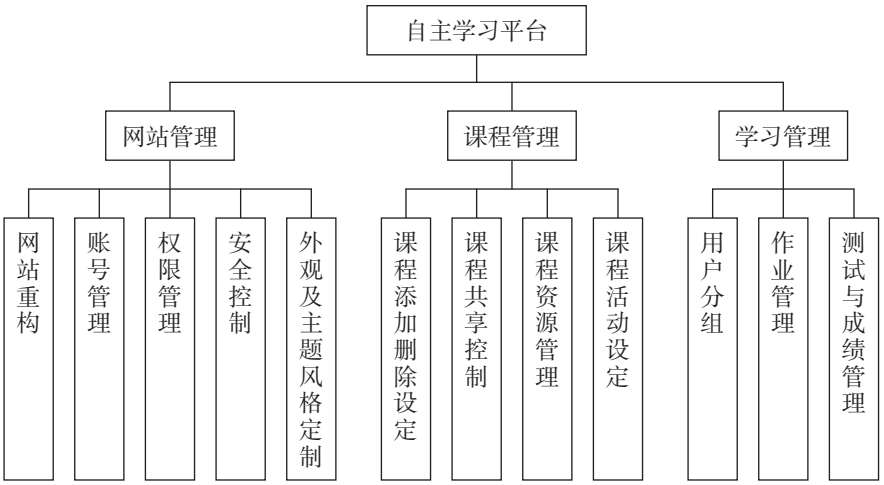
中国版本图书馆 CIP 数据核字(2019)第 251154 号

出版发行 /北京理工大学出版社有限责任公司
社　　址 /北京市海淀区中关村南大街 5 号
邮　　编 /100081
电　　话 /(010)68914775(总编室)
　　(010)82562903(教材售后服务热线)
　　(010)68948351(其他图书服务热线)
网　　址 /http://www. bitpress. com. cn
经　　销 /全国各地新华书店
印　　刷 /三河市天利华印刷装订有限公司
开　　本 /787 毫米 ×1092 毫米　1/16
印　　张 /19. 5　　责任编辑 /高　芳
字　　数 /458 千字　　文案编辑 /高　芳
版　　次 /2019 年 11 月第 3 版　2020 年 6 月第 2 次印刷　　责任校对 /周瑞红
定　　价 /49. 00 元　　责任印制 /李志强

软件技术专业自主学习平台介绍

软件技术专业自主学习平台是一个集合了软件技术专业核心课程的学习管理系统。该平台集合了“JSP 设计与开发”“Android 应用软件开发”“J2EE 技术与应用”等省级精品课程和院级精品课程，提供课程的视频、课件、素材、源码、仿真环境等教学材料，还提供聊天室、论坛等互动交流平台，更提供课程题库等测验环节以方便学习者学习、交流和评测。

平台主要实现了课程管理、作业管理、论坛管理、测验管理、资源管理、互动评价等功能，功能结构图如下。



平台资源列表如下：

平台首页	课程分类	课程标准	课时教案	授课计划	配套教材	配套课件
案例导学	案例视频	案例源码	互动评价	在线实验	仿真环境	自我测验
在线考试	题库	作品展示	微课程	在线问答	聊天室	行业新闻

软件技术专业自主学习平台以社会建构主义教学法为其设计的理论基础。它允许师生或学生彼此间共同思考，合作解决问题。从这些过程中，与他人互动，或与教师互动时，学生很自然就能建立概念，因为他们在交谈时，共同创造出一个可论述的世界，和一个共同架构，在其中可以产生沟通。最终实现“集体智慧”和“集体认知”。

软件技术专业自主学习平台把翻转课堂教学模式和慕课（MOOC）的优点融合在一起，从而革新了传统的教学方式和组织形式，更加有利于提高学生的自主学习能力，培养学生小组协作能力和与人沟通的能力。提出了基于 MOOC 的翻转课堂教学模式，为开展其他课程的翻转课堂教学提供了借鉴价值。

软件技术专业自主学习平台适用于大中专学生进行软件技术专业课程的自主学习，为师生之间的教学交流及资源共享搭建了平台，为从事软件开发的工程人员提供技术支持。

软件技术专业自主学习平台网址为：http://www.cqepc.cn:8887。

该平台不仅可以在普通计算机上访问，还可以通过移动终端进行访问，如需利用此平台开展教学，进行教师权限和学生权限的分配，请联系 10710121@qq.com。

此外，我们还发布了微信公众平台方便用户随时随地进行学习，平台二维码如下。

前　言

为了适应软件企业对学生的要求，满足社会对软件人才的需要，让学生动手开发一个实际项目，在任务中不断动手实践，通过任务驱动学习新的知识，教材基于这一目的进行编写。本书是重庆市示范性高等职业院校重点建设专业（软件技术专业）的特色教材，是“JSP 设计与开发”精品课程的配套教材，是“任务驱动、案例教学、理实一体化”教学方法的载体，突出职业特色和实践特色，侧重于培养学生软件设计、代码编写、软件文档编写规范等能力。

JSP（全称为 Java Server Page）是由 Sun 公司于 1999 年 6 月推出的一种基于 Java 的 Web 开发技术，可以无缝地运行在 UNIX、Linux、Windows 等操作平台上，是目前热门的跨平台动态 Web 应用开发技术。它充分继承了 Java 的众多优势，包括一次编写随处运行的承诺、高效的性能以及强大的可扩展力。特别是结合 Servlet 和 JavaBean 技术，使得 JSP 技术较其他 Web 开发技术有显著的优势。

本教材在编写思想上，以适应高职高专教学改革的需要为目标，以企业需求为导向，充分吸收国外经典教材及国内优秀教材的优点，结合高职院校计算机教育的教学现状，进行内容的组织和编写。

在内容安排上，充分体现先进性、科学性和实用性，尽可能选取最新、最实用的技术，并依照学生接受知识的一般规律，通过设计详细、可实施的项目化案例，帮助学生掌握要求的知识点。

在教材形式上，利用网络等现代技术手段实现立体化的资源共享，为教材配套课程创建专门的网站，并提供题库、素材、录像、课件、案例分析，实现教师和学生在更大范围内的教与学互动，及时解决教学过程中遇到的问题。

本系列教材采用案例式的教学方法，以实际应用为主，理论够用为度。教材中知识点的结构模式为“理论知识介绍→案例（任务）提出→案例要点分析→具体操作步骤→案例总结（理论总结、功能介绍、方法和技巧等）”。

本教材由重庆航天职业技术学院陈磊、徐受蓉老师任主编，重庆师范大学尚晋老师、重庆航天职业技术学院董明、赵叶青、刁绫老师任副主编，教材第 1 章由徐受蓉编写；第 2 章由赵叶青编写；第 3 章由董明编写；第 6 章由尚晋编写；第 4 章、第 5 章、第 7 章、第 9 章、第 10 章由陈磊编写；第 8 章由重庆航天职业技术学院蒋文豪编写；第 11 章由刁绫编写；第 12 章由冯林（重庆航天火箭电子技术有限公司）编写；徐受蓉、刁绫负责全书代码测试。陈磊、董明、赵叶青、冯林完成了本课程的视频录制工作。

本教材配有慕课学习平台：http：//www. cqepc. cn：8887。

编　者

目　　录

第 1 章　JSP 概述

【学习要点】

- Web 程序设计模式与运行原理
- JSP 页面与 JSP 运行原理
- JSP 运行环境的配置
- 集成开发环境简介

1.1　Web 程序设计模式与运行原理

在学习 JSP 编程技术之前，需要对 Web 程序设计模式有所了解。Web 程序的运行方式不同于单机的 Windows 应用程序，本书主要从 Web 服务、浏览器/服务器模式与动态网页技术 3 个方面作简要介绍。

1.1.1　Web 服务器与动态网页

互联网中有数以亿计的网站，用户可以通过浏览这些网站获得所需要的信息。例如，用户在浏览器的地址栏中输入“http://www.sina.com.cn”，浏览器就会显示新浪网的首页，从中可以查看新闻等信息。那么新浪网首页的内容是存放在哪里的呢？新浪网首页的内容是存放在新浪网服务器上的。所谓服务器，就是网络中的一台主机，由于它提供 Web、FTP 等网络服务，因此称其为服务器。

用户的计算机又是如何将存在网络服务器上的网页显示在浏览器中的呢？当用户在地址栏中输入新浪网地址（URL，统一资源定位符）的时候，浏览器会向新浪网的服务器发送 HTTP 请求，这个请求使用 HTTP 协议，其中包括请求的主机名、HTTP 版本号等信息。服务器在收到请求信息后，将回复的信息（一般是文字、图片等网页信息，也就是 HTML 页面）准备好，再通过网络发回给客户端浏览器。客户端的浏览器在接收到服务器传回的信息后，将其解释并显示在浏览器的窗口中，这样用户就可以进行浏览了。整个过程如图 1－1 所示。

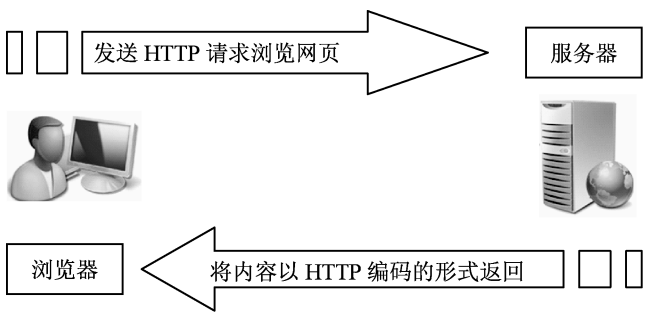


图 1－1　浏览网页的过程

在这个“请求—响应”过程中，如果在服务器上存放的为静态 HTML 网页文件，服务

器就会原封不动地返回网页的内容。如果存放的是动态网页，如 JSP、ASP、ASP. NET 等文件，则服务器会执行动态网页，执行的结果是生成一个 HTML 文件，然后再将这个 HTML 文件发送给客户端浏览器，客户浏览器将其解释为用户见到的页面。

因此，动态网页和静态网页的根本区别在于服务器端返回的 HTML 文件是事先存储好的还是由动态网页程序生成的。静态网页文件里只有 HTML 标记，没有程序代码，网页的内容是事先写好并存放在服务器上的；动态网页文件不仅含有 HTML 标记，而且还含有程序代码，当用户发出请求时，服务器由动态网页程序即时生成 HTML 文件。动态网页能够根据不同的时间、不同的用户生成不同的 HTML 文件，显示不同的内容。

1.1.2 浏览器/服务器结构及其优点

随着网络技术的不断发展，单机的软件程序已经难以满足网络计算机的需求，因此，基于网络的软件架构应运而生。早期常用的网络架构为“客户/服务器”（Client/Server，C/S）模式。使用这种架构编写的软件分为客户端和服务器端两部分，需要分别在客户机和服务器上进行安装。这种模式在用户数据录入等方面很有优势，也降低了系统的通信开销，但是也有一定的缺点，如开发和维护成本较高，可移植性较差等。

互联网的普及使得用于上网浏览的浏览器已经成为操作系统中不可缺少的一部分，浏览器的功能越来越强大，甚至可以取代“客户/服务器”架构的客户端软件，成为统一的客户端。这样，程序员就可以只编写运行在服务器上的软件，浏览器代替 C/S 模式中的客户端软件，客户通过浏览器与服务器端软件进行交互并得到运行结果，这种软件架构就是“浏览器/服务器”（Browser/Server，B/S）模式。B/S 模式主要是利用了不断成熟的 WWW 浏览器技术，结合动态网站制作技术，通过通用浏览器实现了原来需要复杂的专用软件才能实现的强大功能，节约了开发成本，是一种全新的软件系统构造技术。随着互联网的不断发展，B/S 架构已经成为当今应用软件的首选体系结构。

B/S 模式的应用程序相对于传统的 C/S 模式的应用程序来讲无疑是一个巨大的进步，主要优点如下。

1. 开发、维护成本较低

就 C/S 模式的软件而言，当客户端的软件需要升级的时候，所有客户端都必须进行升级安装或者重新安装，而 B/S 模式的软件只需要在服务器端发布，客户端浏览器无须维护，因而极大地降低了开发和维护成本。

2. 可移植性高

C/S 模式的软件，不同开发工具开发的程序，一般情况下互不兼容，主要运行在局域网中，移植困难，而 B/S 模式的软件运行在互联网上，提供了异种网、异种机、异种应用服务的联机、联网服务基础，客户端安装的是通用浏览器，不存在移植的问题。

3. 用户界面统一

C/S 模式软件的客户端界面由所安装的客户端软件所决定，因此不同的软件客户端界面不同，而 B/S 模式的软件都是通过浏览器来使用的，操作界面基本统一。

1.1.3 JSP 与其他 Web 开发技术

在简单介绍了 Web 服务器、动态网页和 B/S 模式的 Web 应用程序结构的优点之后，那

么，哪些技术可用于 B/S 模式的 Web 应用程序开发？目前使用较多的技术有 JSP、ASP、ASP. NET、PHP 等。本节对它们进行简单的介绍和比较。

JSP 全称为 Java Server Pages，是 Sun 公司倡导、多家公司参与、1999 年提出的一种 Web 服务技术标准。它的主要编程脚本为 Java 语言，同时还支持 JavaBeans/Servlet 等技术，利用这些技术可以建立安全、跨平台的 Web 应用程序。JSP 技术具有以下优点。

1. 跨平台性

由于 JSP 的脚本语言是 Java 语言，因此它具有 Java 语言的一切特性。同时，JSP 也支持现在的大部分平台，拥有“一次编写，到处运行”的特点。

2. 执行效率高

当 JSP 第一次被请求时，JSP 页面转换成 Servlet，然后被编译成 *. class 文件，以后（除非页面有改动或 Web 服务器被重新启动）再有客户请求该 JSP 页面时，JSP 页面不再被重新编译，而是直接执行已编译好的 *. class 文件，因此执行效率高。

3. 可重用性

可重用的、跨平台的 JavaBeans 和 EJB（Enterprise JavaBeans）组件，为 JSP 程序的开发提供了方便。例如，用户可以将复杂的处理程序（如对数据库的操作）封装到组件中，在开发中可以多次使用这些组件，提高了组件的重用性。

4. 将内容的生成和显示进行分离

使用 JSP 技术，Web 页面开发人员可以使用 HTML 或者 XML 标记来设计和格式化最终页面。生成动态内容的程序代码封装在 JavaBeans 组件、EJB 组件或 JSP 脚本段中。在最终页面中使用 JSP 标记将 JavaBeans 组件中的动态内容引入。这样，可以有效地将内容生成和页面显示分离，使页面的设计人员和编程人员可以同步进行工作，也可以保护程序的关键代码。

ASP 是 Active Server Pages 的缩写，是微软在早期推出的动态网页制作技术，包含在 IIS（Internet 信息服务）中，是一种服务器端的脚本编写环境，使用它可以创建和运行动态、交互的 Web 服务器应用程序。在动态网页技术发展的早期，ASP 是绝对的主流技术，但是它也存在着许多缺陷。由于 ASP 的核心是脚本语言，决定了它的先天不足，其无法进行像传统编程语言那样的底层操作；由于 ASP 通过解释执行代码，因此运行效率较低；同时由于脚本代码与 HTML 代码混在一起，不便于开发人员进行管理和维护。随着技术的发展，ASP 的辉煌已经成为过去，微软也已经不再对 ASP 提供技术支持和更新，ASP 技术目前处于被淘汰的边缘。

PHP 从语法和编写方式上来看与 ASP 类似，是完全免费的，最早是一个开放源码的小软件，随后逐渐发展起来，是因为越来越多的人意识到它的实用性。Rasmus Lerdorf 在 1994 年发布了 PHP 的第一个版本。从那时起它就飞速发展，在原始发行版上经过无数的改进和完善，现在已经发展到 5.0 版。PHP + MySQL + Linux 的组合是最常见的，因为它们都可以免费获得。但是 PHP 的弱点也是很明显的，例如 PHP 不支持真正意义上的面向对象编程，接口支持不统一，缺乏正规支持，不支持多层结构和分布式计算等。

ASP. NET 是微软继 ASP 后推出的全新的动态网页制作技术，目前最新版本为 . NET5. 0。在性能上，ASP. NET 比 ASP 强很多，与 PHP 相比，也存在明显的优势。ASP. NET 可以使用 C#、VB. NET、Visual J#等语言来开发，程序开发人员可以选择自己习惯或熟悉的语言进行开发。ASP. NET 依托于 . NET 平台先进而强大的功能，从而极大地简化了编程人员的工作

量，使得 Web 应用程序的开发更加方便、快捷，同时也使得程序的功能更加强大，是 JSP 技术的有力竞争对手。

1.2 JSP 页面与 JSP 运行原理

JSP 页面的组成、Web 服务目录和运行原理是读者学习 JSP 的基础，本节对其作简单的介绍。读者在这里了解 JSP 页面和执行原理即可，详细内容将在后续章节中介绍。

1.2.1 案例：第一个 JSP 页面

一个 JSP 页面是由普通的 HTML 标记和 JSP 标记，以及通过“<%”“%>”标记加入的 Java 程序片段组成的页面。JSP 页面按文本文件保存，文件名要符合 JSP 标识符的规定，即文件名可以由字母、数字、下划线或美元符号组成，并且第一个字符不能是数字，文件扩展名为 jsp。用户可以用记事本或其他文本编辑工具，如 EditPlus，来编辑 JSP 文件，文件保存的编码选择 ANSI。

【案例功能】向客户端输出“这是我的第一个 JSP 程序”页面。

【案例目标】掌握 JSP 页面的基本框架结构。

【案例要点】HTML 标识、JSP 语言。

【案例步骤】

案例视频扫一扫

（1）新建一个文本文件，名称为 myfirst，并将后缀名改为 .jsp。

（2）编写 myfirst.jsp 源代码文件。

【源码】myfirst.jsp

```
1  <%@ page contentType="text/html; charset=GB2312" %>
2  <html>
3  <head><title>这是我的第一个 JSP 程序</title></head>
4  <body bgcolor=cyan>
5  <h2>这是我的第一个 JSP 程序</h2>
6  <h3><% out.println("世界你好!"); %></h3>
7  </body>
8  </html>
```

【代码说明】

- 所有黑体标识都是 HTML 标签。
- 在 <%...%> 中嵌入的是 JSP 源码。

（3）将 myfirst.jsp 文件保存到 Tomcat 6.0 安装目录下的 webapps\ROOT 子目录中，本教材 Tomcat 6.0 的安装目录为 D:\Program Files\Apache Software Foundation\Tomcat 6.0。则页面保存目录为 D:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\ROOT，页面运行结果如图 1-2 所示。

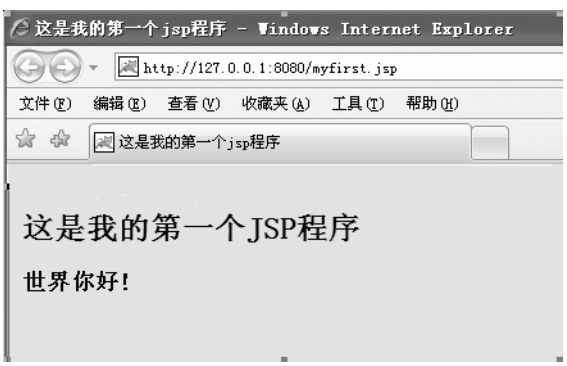


图 1-2 页面运行效果

1.2.2　JSP 运行原理

在上述案例中，用户在客户端浏览器中输入 http://127.0.0.1：8080/myfirst.jsp，浏览器就会显示页面的内容。那么包含 HTML 标记、JSP 标记和 <%...%> 的 JSP 页面是如何显示到客户浏览器中的呢？回答这个问题前，需要了解 JSP 的运行原理。

用户在客户端浏览器中输入 http://127.0.0.1：8080/myfirst.jsp，就会对 Web 服务器上的 myfirst.jsp 页面产生请求。当服务器上的 myfirst.jsp 页面第一次被请求时，JSP 引擎首先转译 JSP 页面文件，形成一个 Java 文件（本质上是一个 Java Servlet 的 Java 文件，关于 Servlet，后续章节还要介绍，在这里，读者可以简单地将其理解为执行在服务器端的 Java 小程序），这个 Servlet Java 文件的文件名是 myfirst_jsp.java，存储在 Tomcat 安装目录的work\Catalina\localhost\...\org\apache\jsp 子目录中，然后 JSP 引擎调用 Java 编译器编译这个文件，形成 Java 的字节码文件 myfirst_jsp.class，存放在相同的目录中；编译完成之后，JSP 引擎就会执行 myfirst_jsp.class 字节码文件，响应客户的请求，执行 myfirst_jsp.class 的结果是发送给客户端一个 HTML 页面。当这个页面再次被请求时，JSP 引擎将直接执行这个编译了的字节码文件来响应客户的请求。当多个用户请求同一个 JSP 页面时，Tomcat 服务器为每个客户启动一个线程，该线程负责执行常驻内存的字节码文件来响应客户请求。JSP 的运行原理如图 1-3 所示。

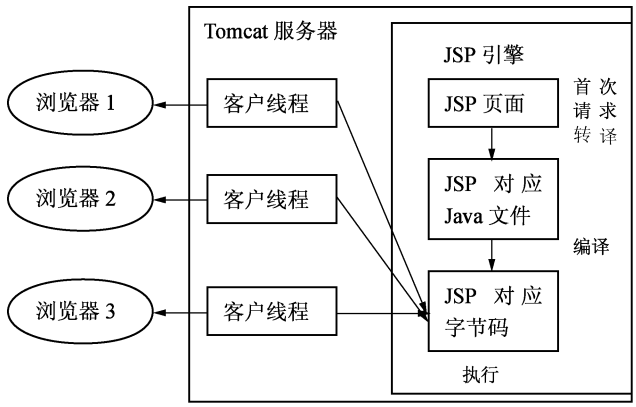


图 1-3　JSP 的运行原理

下面是 JSP 引擎生成的 myfirst_jsp.java 文件的内容（读者可以从 Tomcat 6.0 安装目录的 work\Catalina\localhost\...\org\apache\jsp 目录中找到该文件及其对应的字节码文件）。

```
1  package org.apache.jsp;
2
3  import javax.servlet.*;
4  import javax.servlet.http.*;
5  import javax.servlet.jsp.*;
6  import java.util.*;
7
8  public final class myfirst_jsp extends org.apache.jasper.runtime.HttpJspBase
9      implements org.apache.jasper.runtime.JspSourceDependent {
10
11   private static final JspFactory _jspxFactory =
12 JspFactory.getDefaultFactory();
13
14   private static java.util.List _jspx_dependants;
15
```

```
16    private javax.el.ExpressionFactory _el_expressionfactory;
17    private org.apache.AnnotationProcessor _jsp_annotationprocessor;
18
19    public Object getDependants(){
20      return _jspx_dependants;
21    }
22
23    public void _jspInit(){
24      _el_expressionfactory =
25  _jspxFactory.getJspApplicationContext(getServletConfig().getServletContext()).
26  getExpressionFactory();
27      _jsp_annotationprocessor =(org.apache.AnnotationProcessor)
28  getServletConfig().getServletContext().getAttribute(org.apache.
29  AnnotationProcessor.class.getName());
30    }
31
32    public void _jspDestroy(){
33    }
34
35    public void _jspService(HttpServletRequest request, HttpServletResponse
36  response)
37          throws java.io.IOException, ServletException {
38
39      PageContext pageContext = null;
40      HttpSession session = null;
41      ServletContext application = null;
42      ServletConfig config = null;
43      JspWriter out = null;
44      Object page = this;
45      JspWriter _jspx_out = null;
46      PageContext _jspx_page_context = null;
47
48
49      try {
50        response.setContentType("text/html; charset=GB2312");
51        pageContext = _jspxFactory.getPageContext(this, request, response,
52                  null, true, 8192, true);
53      _jspx_page_context = pageContext;
54      application = pageContext.getServletContext();
55      config = pageContext.getServletConfig();
56      session = pageContext.getSession();
57      out = pageContext.getOut();
58      _jspx_out = out;
59
60      out.write("\r");
```

```
61      out.write("\n");
62
63  String path = request.getContextPath();
64  String basePath =
65  request.getScheme() + "://" + request.getServerName() + ":" + re-
    quest.getServerPort
66  () + path + "/";
67
68      out.write("\r\n");
69      out.write("<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01
70  Transitional//EN\">\r\n");
71      out.write("<html>\r\n");
72      out.write("  <head>\r\n");
73      out.write("  <base href=\"");
74      out.print(basePath);
75      out.write("\">\r\n");
76      out.write("  \r\n");
77      out.write("  <title>这是我的第一个 JSP 程序</title>\r\n");
78      out.write("  </head>\r\n");
79      out.write("  \r\n");
80      out.write("  <body bgcolor=cyan>\r\n");
81      out.write("  <h2>这是我的第一个 JSP 程序</h2>\r\n");
82      out.write("  <h3>");
83  out.println("世界你好!");
84      out.write("</h3>\r\n");
85      out.write("  </body>\r\n");
86      out.write("</html>");
87    } catch(Throwable t){
88      if(!(t instanceof SkipPageException)){
89        out = _jspx_out;
90        if(out != null && out.getBufferSize()!= 0)
91          try { out.clearBuffer(); } catch(java.io.IOException e){}
92        if(_jspx_page_context != null)
    _jspx_page_context.handlePageException(t);
        }
      } finally {
        _jspxFactory.releasePageContext(_jspx_page_context);
      }
    }
  }
```

下面是客户端浏览器看到的源码。

```
1  <%@ page contentType="text/html;charset=GB2312" %>
2  <html>
3  <head><title>这是我的第一个 JSP 程序</title></head>
4  <body bgcolor=cyan>
5  <h2>这是我的第一个 JSP 程序</h2>
6  <h3><% out.println("世界你好!"); %></h3>
7  </body>
8  </html>
```

分析客户端 HTML 代码和服务器端 Java 文件代码，可以看到 out. write（"..."）用于向客户端输出 HTML 代码，JSP 页面对应 Java 文字码文件的主要工作如下。

（1）把 JSP 页面中的 HTML 标记发给客户端浏览器。

（2）负责处理 JSP 页面中的 JSP 标记，并将处理结果发给客户端浏览器。

（3）负责执行“<%”和“%>”之间的 Java 程序，并将执行结果发给客户端浏览器。

1.2.3 JSP、JavaBean 和 Java Servlet 的关系

Java Servlet 是 Java 语言的一部分，它提供了一组用于服务器端编程的 API。习惯上称使用 Java Servlet API 的相关类和方法所编写的 Java 类为 Servlet 类，Servlet 类生成的对象为 Servlet 对象。Servlet 对象可以运行在配置有 JSP 运行环境的服务器上，访问服务器的各种资源，这极大地扩展了服务器的功能。

JSP 是晚于 Java Servlet 产生的，它是为了克服 Java Servlet 的缺点，以 Java Servlet 技术为基础的 Web 应用开发技术标准。JSP 提供了 Java Servlet 的绝大多数优点，是 Java Servlet 技术的成功应用，不过 JSP 只是 Java Servlet 技术的一部分，而不是 Java Servlet 的全部。JSP 可以让 JSP 标记、Java 语言代码嵌入到 HTML 语句中，这样就大大地简化和方便了网页的设计和修改，但 JSP 页面最终会被编译成 Servlet 并执行，以响应客户端的请求。

JavaBean 被 Sun 公司定义为一个可重用的软件组件。实际上 JavaBean 就是一种 Java 类，通过封装属性和方法成为具有某种业务逻辑处理能力的类，它一般负责 Web 应用系统的业务逻辑处理部分。JavaBean 类实例化的对象简称为 Bean。JSP 提供访问 JavaBean 组件的 JSP 动作标记。JSP 动作标记简单、方便，有效地分离了 JSP 页面的表示部分和业务逻辑、数据处理部分，因此使程序设计人员和页面设计人员可以同时工作。

较小规模的 Web 应用可以采用 JSP + JavaBean 模式。在 JSP + JavaBean 模式中，JSP 负责页面的实现、页面预处理和跳转控制，JavaBean 负责业务逻辑和数据处理。对于模式较大的 Web 应用，就需要采用 JSP + JavaBean + Servlet 模式。在 JSP + JavaBean + Servlet 模式中，JSP 负责页面处理（View），JavaBean 负责业务逻辑和数据处理（Model），Servlet 负责预处理和分发页面的请求（Control）。关于这些模式的具体应用将在后续章节中讲述。

1.3 搭建 JSP 运行环境

1.3.1 安装和配置 JDK

Sun 公司提供了一个免费的 Java 软件开发工具包 JDK（Java Development Kit），该工具包